

SPECIFICATION

Docket No. 0544MH-40017

TO ALL WHOM IT MAY CONCERN:

BE IT KNOWN that we, Daniel Brown and Fernando Zapata, residing in the State of Texas, have invented new and useful improvements in a

COMPUTER SECURITY SYSTEM

of which the following is a specification:

CROSS REFERENCE TO RELATED APPLICATIONS

1 The present application claims the benefit of US Provisional application
2 No. 60/187375, filed on March 6, 2000.

BACKGROUND OF THE INVENTION

3 1. Field of the Invention:

4 The present invention relates generally to computer systems, and more
5 specifically to security systems and methods for controlling and authorizing access
6 to computer systems.

7 2. Description of the Prior Art:

8 Security is an important consideration for computer systems that grant
9 access to multiple users. This is especially true when access can be obtained
10 from outside a physically restricted area, such as systems generally available
11 over the internet. As systems become more complex, and access to data needs
12 to be more widely distributed, security systems tend to become more complex.

13 Such issues arise in the context of different businesses sharing data and
14 processes over distributed computer systems. It becomes important that security
15 can be administered from more than one location, by more than one
16 administrator. Further, due to different companies doing business in different
17 ways, security systems intended for use by different business must be flexible,
18 and able to accommodate different security implementations.

1 Present security systems are often cumbersome, and do not have the
2 desired flexibility. It would be desirable to provide a computer security system
3 that was flexible, extendable, and allowed multiple administrators to operate
4 concurrently to provide needed security. It would be desirable that such a
5 system allows administrators to define additional types of security, and supply
6 security for additional types of objects, than are originally provided for.

1990		1991		1992		1993		1994		1995		1996		1997		1998		1999		2000		2001		2002		2003		2004		2005		2006		2007		2008		2009		2010		2011		2012		2013		2014		2015		2016		2017		2018		2019		2020		2021		2022		2023		2024		2025		2026		2027		2028		2029		2030		2031		2032		2033		2034		2035		2036		2037		2038		2039		2040		2041		2042		2043		2044		2045		2046		2047		2048		2049		2050		2051		2052		2053		2054		2055		2056		2057		2058		2059		2060		2061		2062		2063		2064		2065		2066		2067		2068		2069		2070		2071		2072		2073		2074		2075		2076		2077		2078		2079		2080		2081		2082		2083		2084		2085		2086		2087		2088		2089		2090		2091		2092		2093		2094		2095		2096		2097		2098		2099		2100	
1990	1991	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018	2019	2020	2021	2022	2023	2024	2025	2026	2027	2028	2029	2030	2031	2032	2033	2034	2035	2036	2037	2038	2039	2040	2041	2042	2043	2044	2045	2046	2047	2048	2049	2050	2051	2052	2053	2054	2055	2056	2057	2058	2059	2060	2061	2062	2063	2064	2065	2066	2067	2068	2069	2070	2071	2072	2073	2074	2075	2076	2077	2078	2079	2080	2081	2082	2083	2084	2085	2086	2087	2088	2089	2090	2091	2092	2093	2094	2095	2096	2097	2098	2099	2100																																																																																																															

1 In accordance with the present invention, a security system for a computer
2 system provides one or more security domains. Access to assets registered to the
3 security system is controlled by rights and privileges. Rights are derived from roles,
4 and each user is assigned one or more roles. Privileges are attached to assets,
5 and an appropriate combination of rights and privileges is required before a user is
6 granted the specified type of access to the asset.

BRIEF DESCRIPTION OF THE DRAWINGS

1 The novel features believed characteristic of the invention are set forth in the
2 appended claims. The invention itself however, as well as a preferred mode of use,
3 further objects and advantages thereof, will best be understood by reference to the
4 following detailed description of an illustrative embodiment when read in
5 conjunction with the accompanying drawings, wherein:

6 Figure 1 is a block diagram of a system for providing security to a computer
7 system;

8 Figure 2 is a diagram illustrating the structure of a computer security system
9 in accordance with the present invention;

10 Figures 3 and 4 are diagrams illustrating domain relationships in accordance
11 with a preferred embodiment;

12 Figure 5 is a diagram illustrating preferred steps in granting access and
13 authorization to a user;

14 Figure 6 is a diagram illustrating the use of roles within a preferred security
15 system;

16 Figure 7 is a diagram showing the relationships between rights and
17 privileges in a preferred embodiment;

18 Figures 8 and 9 illustrate preferred use of access control lists in accordance
19 with the preferred embodiment;

1 Figure 10 is a diagram showing the use of bi-directional transfer of
2 privileges; and

3 Figures 11 and 12 are block diagrams illustrating preferred user
4 authorization methods.

Patent Application US utility.doc

DESCRIPTION OF THE PREFERRED EMBODIMENT

1 It will be understood by those skilled in the art that the following description
2 can easily be implemented on numerous different underlying systems. The
3 described system describes a particular set of techniques and methods for granting
4 users access to various files, executables, and other system assets available on
5 the system being protected. The described security system and method does not
6 necessarily provide complete system security, but can be supplemented by other
7 products widely available in order to provide complete security. As will be
8 appreciated by those of ordinary skill, the description below indicates where and
9 how it is to be implemented on any desired system.

10 Figure 1 is a block diagram of a security system implemented in accordance
11 with a preferred embodiment of the invention. Figure 1 shows the architecture of
12 the preferred system, with particular focus on the security service. A first host
13 machine 10 includes a web server application 12 and a servlet engine 14.
14 Application logic 16 that resides in the servlet engine 14 makes in-process calls
15 to the security API where applicable. The security service communicates with a
16 database 18, which can be an Oracle database, via JDBC link 20.

17 A second host machine 22 contains the application 24 functionality to which
18 access is desired. The application 24 in turn relies on the security service 26 for
19 access control information. In one embodiment, the BO Server 24 is a C++
20 engine, and communicates with Security 26 using a CORBA server called the
21 `DNA Bridge 28. The DNA Bridge 28 is responsible for sending raw

1 permissibility data to the BO Server 24, which then handles the actual
2 enforcement.

3 The described security model is centered on the concept of a **domain 29**,
4 shown in Figure 2. A domain is an administrative and access control boundary
5 around a collection of security entities. These entities consist of:

- 6 1. **members 30, 31**, which are users who have authenticated themselves to the
7 domain;
- 8 2. **groups 32**, which are collections of members used to facilitate administration;
- 9 3. **roles 34**, which represent the responsibilities that members may assume;
- 10 4. **registered assets 36**, which are resources that the system is responsible for
11 protecting;
- 12 5. **ACLs (Access Control Lists) 38**, which define the privileges that roles
13 should have when accessing assets.

14 Domains are used to provide a security “sandbox” for members. The
15 “sandbox” controls *what* members may do to *which* assets during a given session
16 with the system. The system has many domains, some or all of which may be in
17 use at any given time. A domain can be mapped to any entity external to
18 Security, but many applications have only found the need to map a domain to a
19 business/company.

20 Most entities reside within a single domain for the duration of a session, but
21 there exist some special entities that have visibility across multiple domains.
22 One of these, the universal security administrator 40 (also called the “super

user”), is a special member who is allowed to administer the entire security model, including all of the entities within any domain. Another special entity is the role. Security uses roles to implement declarative and programmatic security. There are two kinds of roles in the system:

1. **domain roles 34**, which are visible only within a single domain; and
2. **universal roles 42**, which are visible across all domains.

Universal roles 42 represent user responsibilities that are commonly accepted and understood by several collaborating domains. They exist so that the workflow of a shared application is consistent for -- and understandable by -- all of its users. Domain roles 34 have meaning only within the domain in which they are defined.

The domain to which a user authenticates himself at the beginning of a session dictates the roles that he may utilize. The following example illustrates:

1. Let domains be “Acme Computers” and “Beta Bank”
2. Let member be “jsmith”
3. Let universal roles be “Administrator” and “Purchaser”
4. Let domain role for Beta Bank be “Teller”
5. Let domain role for Acme Computers be “Assembler”
6. Let jsmith be assigned to both domains
7. Let jsmith be assigned to roles Administrator, Purchaser, and Teller

If jsmith logs into the Acme Computers domain, his active role set consists of:

Administrator, Purchaser

If jsmith logs into the Beta Bank domain, his active role set consists of:

1 Administrator, Purchaser, Teller

2 The member jsmith does not have the Assembler role when he logs into
3 the Acme Computers domain because he has not been granted that role.

4 Two domains may be joined by a **trust relationship**. A trust relationship
5 determines how privileges may be delegated from one domain to another. Trust
6 may be unidirectional or bi-directional, as shown in Figure 3. For example,
7 Domain ABC 50 has a unidirectional trust relationship with Domain DEF 54 (ABC
8 trusts DEF), which implies that privileges on ABC's assets may be delegated
9 from ABC to DEF. ABC 50 has a bi-directional trust relationship with XYZ 52,
10 which implies that privileges on ABC's assets may flow to XYZ, and vice-versa.

11 A domain may also own another domain, i.e. be responsible for its
12 creation and destruction. As shown in Figure 4, a **parent-child** topology is used
13 to represent domain ownership and a **peer-to-peer** topology is used when there
14 is a lack of ownership between domains.

15 The trust and ownership concepts may be combined. For example, a
16 group of domains may be connected together in a parent-child topology, where
17 each connection is also a unidirectional trust. This implies that privileges may be
18 delegated from the top-most domain down to the lower-level domains. It also
19 implies that each domain has created the domain(s) beneath it.

20 The model described in the previous paragraph can be illustrated by an
21 example. Assume a customer uses this model for its catalogs and categories.
22 The customer creates subsidiary domains and then gives each subsidiary access

1 to a subset of its catalogs and categories. The subsidiaries in turn create
2 distributor domains and then give the distributors access to a further subset of
3 the catalogs and categories. Support for additional domain relationship and
4 topologies will be added as the need arises.

5 The “**security principals**” that are associated with a user – consisting of
6 domains and roles -- determine what the user has access to. As shown in Figure
7 5, security principals are acquired after a user is authenticated. Authentication
8 establishes the user's identity, which can then be used to obtain relevant
9 information about the user's involvement in the business domain (such as what
10 company he is employed with and what portal he has logged into). These
11 “**business principals**” are subsequently used to derive the security principals
12 that should be in effect for the member. The system uses these security
13 principals to make authorization decisions.

14 As indicated in Figure 6, **roles** 62 may be granted directly to a member
15 63, or they may be granted 64 to a group 66. Any member who is assigned to
16 the group will in turn receive the privileges enjoyed by the group roles 64. The
17 relationship between members and roles is determined dynamically at login time.
18 A member 63 may be assigned to multiple roles, but only a subset of them may
19 be usable within a given domain. For example, the member 63 in Figure 6 has
20 been granted Role 1 62 and Role 2 64, but Domain A 68 only allows him to use
21 Role 1 62, whereas Domain B 70 allows him to use both Roles 1 and 2. Whether
22 or not a role is usable in a domain depends on the type of role (i.e. universal role
23 or domain role).

As shown in Figure 7, the roles that a member has been granted ultimately determine what assets 74 the member can access. Access control decisions are made by combining **rights** with **privileges**:

1. **rights** 76 are attached to roles 72 and override privileges on assets. For example, the universal security administrator (super user) 40 has special rights which enable him to administer any asset in the system regardless of the privileges that have been specified on the asset. The universal security administrator is an example of a right that is defined by the Security system, but rights may also be defined by an application. Applications that control workflow based upon a particular role are implicitly attaching rights to that role.

2. **privileges** 78 are attached to assets 74 and are used to grant fine-grained access to an asset

A single privilege identifies what operation may be performed by what role on which asset. The mechanism for attaching privileges to assets 74 is the **access control list (ACL) 80**. An ACL 80 contains a series of **access control entries (ACEs) 82**, each of which contains a domain identifier 84, a role identifier 86, and one or more privileges 88, as shown in Figure 8. The domain identifier 84 and role identifier 86 are the security principals that may be associated with one or more members. An ACE 82 contains both because a privilege for a role must be scoped by the domain (e.g. an Admin may be able to perform some operation in one domain, but may be prevented from doing so in a different domain).

1 A privilege allows the domain/role combination to perform an **operation** on
2 the asset. As shown in Figure 9, the basic operations consist of **read** and **write**.
3 Other operations may be available, but these are dependent upon the type of
4 asset being administered.

5 In addition to privileges, a domain/role may be granted ownership over a
6 particular operation on the ACL. An owner of an ACL is allowed to modify it
7 within certain limits:

- 8 1. if a domain/role owns an operation on an ACL, a member who possesses
9 that domain/role principal combination may grant the operation to the
10 *same* role within a different domain.

11 For example, in Figure 9, a member who is in Domain ABC and has Role Foo
12 owns the write operation on the ACL. This member effectively becomes an
13 administrator of the ACL and is therefore allowed to grant the write operation to
14 Role Foo in Domain DEF. He is also allowed to grant Domain DEF/Role Foo
15 ownership of the write operation.

16 ACL ownership and administration are the mechanisms by which privileges
17 are transferred between domains. In a unidirectional transfer of privileges, the
18 capabilities of a role tend to diminish the further you move away from the "home"
19 domain (i.e. where the asset is created). This is because an administrator may
20 never pass on more privileges than he himself has. At most, the role capabilities
21 would remain constant across all of the domains, but in practice this would not be
22 likely.

1 In a bi-directional transfer of privileges, as shown in Figure 10, the capabilities
2 of a role would vary across domains according to the asset in question.

3 A **registered asset** is a resource that the security system is responsible for
4 protecting. Registered assets are classified according to their **asset type**, which
5 determines how assets should be identified and what operations may be
6 performed on them. One possible list of basic asset types includes price group,
7 price template, catalog, category, product group, and URL. This list would be of
8 use in deploying an ordering system, or similar enterprise. New asset types may
9 be defined at deployment time, as described below.

10 An asset type defines both **meta-level** and **instance-level** operations. Meta-
11 level operations are those that are performed without an instance, for example
12 the create operation can be invoked for a Price Template, but it does not apply to
13 an instance because the instance does not yet exist. Instance-level operations,
14 such as read, write, and delete, are those that are performed on an explicit
15 instance, such as deleting a Price Template, which requires a specific instance to
16 delete.

17 Individual assets are identified via a name known as a **moniker**. Monikers
18 are alphanumeric strings that conform to a predetermined format as defined by
19 the asset type. Monikers may be hierarchical in nature, and they may be defined
20 in terms of regular expressions. For example, an asset of type URL may have
21 the following as a moniker:

22 /RhythmAuthor/Content/Pricing/CategoryList.jsp

1 The parent asset would be as follows:

2 /RhythmAuthor/Content/Pricing

3 The preferred system supports two modes of authorization: decision-based
4 and entitlement-based. The main difference between the two modes is the
5 information that is returned by the security system. In decision-based mode,
6 illustrated in Figure 11, a simple yes/no answer is returned to the client.
7 Decision-based authorization is defined by the following sequence:

- 8 1. An initiator 90 sends a request to a resource manager 92. The request
9 consists of an asset, an operation to perform on that asset, and the
10 initiator's principals (role and domain).
- 11 2. The request is intercepted by an access enforcement function 94 that lives
12 inside of the resource manager 92.
- 13 3. The access enforcement function 94 forwards the request information to
14 the access decision function 96.
- 15 4. The access decision function 96 renders a yes/no judgment on the
16 request and returns this decision to the access enforcement function 94.
- 17 5. The access enforcement function 94 takes appropriate action (for
18 example, allow request to proceed, redirect, send error message, etc.)
19 based upon the recommendation provided. Access is allowed to the asset
20 98 only if the initiator's principals allow it.

21 In entitlement-based mode, illustrated in Figure 12, a collection of objects 99 is
22 returned to the client. Entitlement-based authorization is defined by the following
23 sequence:

- 24 1. An initiator 90 sends a read request to a resource manager 92. The
25 request consists of query criteria (e.g. all computer peripheral-related
26 categories) and the initiator's principals.
- 27 2. The request is intercepted by an access enforcement function 94 that lives
28 inside of the resource manager 92.
- 29 3. The access enforcement function 94 forwards the request information to
30 the query modification function 100.

1 4. The query modification function 100 adds security-related criteria to the
2 request and executes a business query 102.

3 5. The results of the query are returned to the access enforcement function
4 94.

5 6. The access enforcement function 94 takes appropriate action (e.g. returns
6 query results to the initiator, redirects, etc.)

7 The objects that are returned conform to the initiator's application-specific
8 criteria as well as some security-specific criteria. The query results contain only
9 the objects that the initiator has read access to, which may be a subset of those
10 requested.

11 A preferred embodiment allows implementers to define new asset types. The
12 steps for defining a new asset type are as follows:

13 1. Create a new row in the SEC_ASSET_TYPE table. Supply values for
14 columns NAME, DESCRIPTION, and FORMAT. The latter must be a
15 valid regular expression that defines a unique moniker format (i.e. naming
16 format for the string used to identify the asset). The uniqueness
17 requirement exists because when a request is made for an asset, the
18 moniker is supplied as an argument. In order for security to match the
19 asset with its correct type, the moniker argument is compared against
20 each naming format. The asset type is discovered as soon as an exact
21 match is made. For example, the moniker
22 "BLM:CM.Catalog.Catalog:[2FA4A958AA8311D4985A00508BD626C1]"
23 cannot be mistaken as a moniker for a URL asset type because it will not
24 match the URL format "[A-Za-z0-9/._+]{0,1}.*)".

25 2. Determine what operations should apply to the new asset type. If these
26 operations do not already exist in the SEC_OPERATION table, go ahead
27 and add them. Supply values for columns NAME, DESCRIPTION, and
28 META_FLAG. The value for the latter column should be "false" if the
29 operation is instance-level, and "true" if it is meta-level.

30 3. Associate the operations with the new asset type by adding entries to the
31 SEC_ASSET_OPERATION table. This table is merely a link table to
32 facilitate many-to-many relationships between asset types and operations.

33 The above description sets forth the mechanism and steps of the preferred
34 security system. Although the terms used are relatively self-explanatory, the
35 following list will assist those skilled in the art in understanding the description:

- 1 **Access Decision Function (ADF):** A process that renders an authorization
2 decision (yes/no). An ADF is internal to the security system.
- 3 **Access Enforcement Function (AEF):** A process that reacts to an
4 authorization decision. An AEF typically controls workflow and may be internal or
5 external to the security system.
- 6 **Access Entitlement Function:** A process that performs queries and limits the
7 results to only those objects that an initiator has read access to.
- 8 **Active Principal Set (APS):** A grouping of principals for a particular member.
9 The APS is determined dynamically after a user is authenticated.
- 10 **Anonymous User:** A user whose identity has not been verified by the system.
11 An anonymous user is considered untrustworthy.
- 12 **Asset:** A resource that is eligible for protection by the Security system. It is an
13 actual physical file, application, object, or set of data.
- 14 **Asset Type:** A category of resources having similar security characteristics. An
15 Asset Type is associated with one or more operations, and an operation is
16 associated with one or more asset types.
- 17 **Authentication:** The process of determining the identity of a user.
18 Authentication ensures that users are who they claim to be, but says nothing
19 about their rights to perform an operation on a registered asset.
- 20 **Authenticated User:** A user whose identity has been verified by the system.

Authorization: The process of determining whether a user has the right to perform an operation on a registered (protected) asset. Authorization may occur in two modes:

1. **decision mode**, in which an initiator makes a request to perform an operation on an asset and a yes/no response is furnished, or
2. **entitlement mode**, in which an initiator makes a request to perform the read operation on a set of assets, and a filtered subset of the assets is furnished. The filtered subset contains only those assets that the initiator is allowed to read.

Credential: An item that a user supplies to the system to prove his/her identity. A credential can consist of something the user knows (e.g. password) or has (e.g. digital certificate).

Group: A named collection of members who have similar characteristics.

Initiator: A client that makes a request of the security system. An initiator may be a user or an application.

Member: An account that is associated with an authenticated user. A member is associated with exactly one Person (from Profile Management), and a Person is associated with zero or one member.

Operation: An action or procedure that may be performed upon an asset. An operation is specific to an asset type.

Policy: A set of rules that describe the ability of various users to access the resources within an application. A policy defines the principals that have the right to perform certain operations upon one or more assets.

Principal: A name or identifier that the system uses to make authorization decisions. A principal is a necessary component of a policy. A principal may be associated with zero or more members, and a member may be associated with zero or more principals. The association between a principal and a member is determined dynamically.

Principal Acquisition: The process by which one or more principals are bound to a member to form the Active Principal Set. Principal acquisition occurs after authentication.

Principal Type: A category of principals having similar characteristics. Each principal among those of the same type must be unique. The principal types are: role, company, portal.

Registered Asset: A logical representation of an actual system resource that is under the protection of the Security system.

Authorization Request: An inquiry that an initiator makes of the security system, consisting of an asset, an operation, and an active principal set.

Role: An abstract representation of a user's function or ability.

User: A person or entity that interacts with the application.

User Registration: A process by which a user supplies information about himself and thereby becomes eligible for greater access to the resources within

1 an application. The product of the user registration process is a person and a
2 member (account).

3 The described system allows for easy group administration of the overall
4 security system. Because of transfer of privileges, various levels of
5 administration can be passed down the line. For example, a single super user
6 can designate other administrators having various degrees of administrative
7 authority, so that various administrative functions can be distributed among
8 numerous administrators. Each administrator can delegate all or a portion of her
9 administrative rights as desired.

10 Also, because bi-directional transfer of privileges is allowed, same-level
11 administrators at different companies can cooperate to effectively administer a
12 combined system. This allows administrators to grant privileges for those
13 domains, or portions of a domain, that they are responsible for or familiar with. In
14 this manner, it is not necessary that a single super user is tasked with all final
15 security responsibility.

16 Because a customer can define additional resource types, and additional
17 types of security to be applied to them, the system is more flexible than most
18 available security systems. When combined with the ability for separate domains
19 to be treated together, with individual security within a domain, a truly generic
20 and flexible security system is provided.

21 While the invention has been particularly shown and described with
22 reference to a preferred embodiment, it will be understood by those skilled in the art

- 1 that various changes in form and detail may be made therein without departing from
- 2 the spirit and scope of the invention.

3

Patent Application US utility.doc